US-PAT-NO:	5819160
DOCUMENT-IDENTIFIER:	US 5819160 A
TITLE:	Programmable radio subscription system for receiving selectively defined information
DATE-ISSUED:	October 6, 1998

US-CL-CURRENT: 455/45, 455/186.1, 455/3.04

APPL-NO: 08/ 715971
DATE FILED: September 18, 1996

US Patent No. - PN (1):

5819160

Brief Summary Text - BSTX (11):

In accordance with the present invention a subscriber remotely connects to a subscription control system to selectively define or edit one or more playlists. A playlist is a collection of one or more items of audio information such as music selections, news stories, and literary audio works. The remote connection is preferably accomplished via a data network, such as the Internet, in which the subscriber uses a computer equipped with a modem to interact with the subscription control system. The remote connection may also be accomplished via a voice telephone network, in which case the subscriber uses a touch-tone telephone or the like to interact with the subscription control system.

Brief Summary Text - BSTX (13):

If the subscriber desires to include news selections as part of a playlist being defined, the subscriber may select one or more predefined keywords from the subscription content database to identify a news topic for inclusion in the playlist. For example, the subscriber may select the keyword "BASEBALL" if news about baseball is desired or "NEWS: FRANCE" if the subscriber is interested in news relating to France. Optionally, the subscriber may directly key-in or enter one or more new keywords to identify a particular news topic in lieu of selecting an existing keyword. When the playlist is complete, the playlist ID code is associated with the subscriber ID code, and the playlist is stored in subscriber data memory. The subscriber may then define another playlist, edit the content of an existing playlist or delete an existing playlist.

Detailed Description Text - DETX (9):

The operation of the system of the present invention is generally controlled by a control program stored in a memory 34 of the main control unit 16, and executed through operation of the main control unit. This control program may by way of example consist of multiple integrated program modules, with each module bearing responsibility for operatively controlling one or more functions of the control unit 16. For example, one program module may control the creation of a playlist by a subscriber remotely connected to the main control unit, while another program module may control the assembly of a particular playlist for transmission to a particular subscriber. In effect, each program module is a control program dedicated to a specific function or set of functions of the main control unit 16. Two main modules of the control program of the present invention are described below in connection with FIGS. 2 and 3. The first program module, described in FIG. 2, is executed by the main control unit 16 and controls the configuration of new playlists and the editing of existing playlists by a subscriber remotely connected to the main control unit 16. The second program module, described in FIG. 3, is executed by the main control unit 16 and the radio control unit 26, and controls the assembly, transmission, and playback of a requested playlist to a subscriber.

Detailed Description Text - DETX (13):

If no new playlists are to be created at test 106, the main control unit 16 proceeds to test 116 where the subscriber is prompted whether any existing playlists are to be edited. If an existing playlist is to be edited at step 118 then the subscriber selects a playlist to be edited from a group of existing playlists associated with that subscriber's subscriber ID code. The selected playlist is then retrieved from the subscriber data memory 20 by main control unit 16. At step 120, the subscriber edits the selected playlist by removing or modifying selections and/or news keywords defined in the playlist and/or by adding new selections and/or news keywords that are, for example, retrieved from the subscription content database 18. The subscriber may also delete the selected playlist at step 120. At step 122, the updated playlist is stored in the subscriber data memory 20 and main control unit 16 returns to test 116 and prompts the subscriber as to whether another existing playlist is to be edited. If no playlists are to be edited at test 116, then the main control unit 16 ends the first program module at step 118. Thus, the first program module enables a subscriber to remotely define and select content of one or more playlists for transmission to the subscriber's digital radio 12.

US-PAT-NO: 5892915

DOCUMENT-IDENTIFIER: US 5892915 A

TITLE: System having client sending edit commands to server during transmission of continuous media from one clip in play list for editing the play list

DATE-ISSUED: April 6, 1999

US-CL-CURRENT: 709/219, 725/116, 725/93

APPL-NO: 08/ 851560

DATE FILED: May 5, 1997

PARENT-CASE:

RELATED APPLICATIONS

The present application is a continuation of provisional application Ser. No. 60/044,948 filed Apr. 25, 1997 by Dinesh Venkatesh, Wayne W. Duso, John Forecast, Uday Gupta, Uresh K. Vahalia, and Dennis P. J. Ting, entitled "Raid Striping, Client-Server Protocols, and Failover Services for a Video File Server."

Abstract Text - ABTX (1):

A protocol and interface provides continuous play over multiple clips for extended periods of time, allows a play-list to be edited dynamically after being given to the video server and during playback of clips in the play-list, allows some notion of "current time" to be used during the streaming of continuous media data, and supports features of the "Louth Automation" video disk communications protocol. Preferably, the client application first creates a session with a play-list containing a fixed number of entries; the number should be as small as possible consistent with the client's requirements. The client edits this play-list by appending the first few clips and then starts the session playing. Each time transmission of video data of a clip is completed, the clip is removed from the head of the play-list, all other clips are moved down, and a callback is issued to the client with the current, updated, play-list. A callback is also issued with the updated play-list to acknowledge each edit command. Preferably, there is a limit as to how close to air-time a clip normally may be deleted or new material inserted, in order to ensure continuity of transmission of the video stream of each clip. To allow live break-ins or other "emergency" operations, however, the session may be paused and later resumed and subsequent clips may be "trimmed" to reduce their play times to recover the time lost to the break-in.

US Patent No. - PN (1):

5892915

TITLE - TI (1):

System having client sending edit commands to server during transmission of continuous media from one clip in play list for editing the play list

Brief Summary Text - BSTX (17):

The present invention provides a client-server protocol and interface for providing broadcast playback functionality. In particular, the protocol and interface easily provides continuous play over multiple clips for extended periods of time, allows a play-list to be edited after being given to the video server and during playback of clips in the play-list, allows some notion of "current time" to be used during the streaming of continuous media data, and supports features of the "Louth Automation" video disk communications protocol. The protocol and interface permits an external application to create and manipulate a dynamic play-list; as clips finish playing they are discarded from the play-list and new clips may be appended to or inserted into the play-list at selected positions (subject to server-imposed constraints).

Brief Summary Text - BSTX (18):

In a preferred mode of operation, the client application first creates a session with a play-list containing a fixed number of entries; the number should be as small as possible consistent with the functions that the client wishes to perform. The client edits this play-list by appending the first few clips and then starts the session playing. Each time transmission of video data of the clip is completed, the clip is removed from the head of the play-list and all other clips are moved down. A callback is issued to the client with the current, updated, play-list. At any time while the video session is playing, edit commands may be issued to insert or delete new continuous media into or from the play-list of the video session. Preferably, there is a limit as to how close to air-time a clip normally may be deleted or new material inserted, in order to ensure continuity of transmission of the video stream of each clip. To allow live break-ins or other "emergency" operations, however, the session may be paused and later resumed and subsequent clips may be "trimmed" to reduce their play times to recover the time lost to the break-in.

Detailed Description Text - DETX (153):

The stripe set list associated with each clip, for example, includes a doubly-linked list of entries, and each entry includes a starting stripe set number, an ending stripe set number, and a value indicating the number of data blocks included in the terminal stripe set. Therefore, each entry in the list represents in sequence data blocks beginning in the initial stripe set, continuing in any intermediate stripe set, and ending in the terminal stripe set, and including in the terminal stripe set only the indicated number of data blocks. The stripe set list for each clip can therefore easily be edited by linking and unlinking entries.

Detailed Description Text - DETX (157):

As shown in FIG. 27, the video file server maintains an active client list 301 in order to manage the servicing of client requests. The active client list 301 is essentially a directory to blocks of information about maintaining respective isochronous data streams to the active clients. Such a block of information 302 includes a client identifier 303 identifying the client to which the block of information is relevant, a stream server identifier 304 identifying a stream server assigned to service the client, a play list 305 of clips that are transmitted in sequence to the client, and a current play position 306 in the play list and in the clip currently being played. The play list 305, for example, is a doubly-linked list including, at the head of the list, the clip identifier of the clip currently being transmitted to the client. The video file server responds to a client request for scheduling additional clips by inserting corresponding clip identifiers to the tail of the play list. The video file server responds to a client request to edit its schedule of clips by linking or unlinking corresponding clip identifiers to or from the client's play list.

Detailed Description Text - DETX (332):

To solve these problems, CMFAP has been extended to process a set of commands from the client for providing broadcast playback functionality. In particular, the extended protocol and interface easily provides continuous play over multiple clips for extended periods of time, allows a play-list to be edited after being given to the video server and during playback of clips in the play-list, allows some notion of "current time" to be used during the streaming of continuous media data, and supports features of the "Louth Automation" video disk communications protocol. The extended protocol and interface permits an external application to create and manipulate a dynamic play-list; as clips finish playing they are discarded from the play-list and new clips may be appended to or inserted into the play-list at arbitrary points (subject to server imposed constraints).

Detailed Description Text - DETX (345):

status--status reply; e.g., successful, session in wrong state, insufficient bandwidth, internal communications failure, requested clip missing, requested clip empty, bad endpoint, invalid session handle, invalid clip handle, unsupported operation, insufficient internal resources, bandwidth of requested clip is inconsistent with bandwidth requested when the play-list was created, disk I/O error, network I/O error, generic failure, clip already in use for incompatible operation, attempt to edit too late

Detailed Description Text - DETX (348):

Next, in step 423 of FIG. 34, the client receives the session handle, and uses it to send one or more "edit session" commands to the video file server to add one or more clips to the play-list. Each such edit command may manipulate the state of a single clip within the play-list by adding a new clip or deleting an existing clip within the play-list. A format for such a play-list edit command is:

Detailed Description Text - DETX (358):

status--operation status; may indicate that an attempt to edit a clip within a play-list was made too late to maintain continuous playback

Detailed Description Text - DETX (361):

VAPPeditop.sub.-- t is an enumerated type which defines edit operations on a play-list:

Detailed Description Text - DETX (367):

In response to each "edit session" command, in step 424, the video file server adds a clip to the play-list, and returns to the client the new version of the play-list.

Detailed Description Text - DETX (384):

At any time while the video session is playing, edit commands may be issued to delete or insert new material from or into the play list of the video session. For example, as shown in steps 427 and 428 of FIG. 35, the client may respond to the callback from the server when transmission from a clip is completed by sending one or more "edit session" commands to the server to add additional clips to the play-list. The client may also send "edit session" commands to the video file server during playback of the session to remove clips from the play-list. The video file server responds in step 429 by dynamically revising the play-list during broadcast of the clip at the head of the play-list. Preferably, there is a limit as to how close to broadcast time a clip normally may be deleted or new material inserted, in order to ensure continuity of transmission of the continuous media stream of each clip.

Detailed Description Text - DETX (396):

As seen in the state diagram of FIG. 36, edit session commands can be issued whenever the session exists; i.e., in the session created state, the session playing state, and the session paused state. Edit commands delete or insert new material from or into the play-list during the session playing state without causing an interruption of the broadcast transmission. To ensure continuity of broadcast transmission during each clip, however, it is desirable to set a limit as to how close to air-time a clip normally may be deleted or new material inserted. If this limit would not be met, the edit command is rejected. To allow live break-ins or other "emergency" operations, however, the session may be paused and later resumed and subsequent clips may be "trimmed" to reduce their play times to recover the time lost to the break-in.

Detailed Description Text - DETX (398):

In step 453, the controller server checks whether it is in the "session playing" state for the session identified by the "edit session" command. If not, the possibility of interrupting broadcast transmission does not exist, and execution branches to step 454 to edit the play-list for the session. In step 455, the controller server transmits to the client the edited play-list to acknowledge completion of the "edit session" command, and processing of the "edit session" command is finished.

Detailed Description Text - DETX (399):

If the controller server finds in step 453 that the session is playing, then execution continues to step 456 to check whether the requested edit is too close to air time. In step 456, the controller server computes the difference in time (.DELTA.T) between the earliest time of continuous media to be added to or deleted from the play-list, and the current time. Then, in step 457, the controller server checks whether this difference in time (.DELTA.T) is less than a certain minimum time (TMIN). If not, then execution branches from step 457 to step 454 to edit the play-list. Otherwise, execution continues to step 458. In step 458, the controller server returns to the client an error code indicating that the edit is too close to broadcast time to avoid an interruption, and processing of the "edit session" command is finished.

Detailed Description Text - DETX (457):

In view of the above, there has been described a client-server protocol and interface for providing broadcast playback functionality. The protocol and interface easily provides continuous play over multiple clips for extended periods of time, allows a play-list to be edited after being given to the video server and during playback of clips in the play-list, allows some notion of "current time" to be used during the streaming of continuous media data, and supports features of the "Louth Automation" video disk communications protocol. The protocol and interface permits an external application to create and manipulate a dynamic play-list; as clips finish playing they are discarded from the play-list and new clips may be appended to or inserted into the play-list at selected positions (subject to server-imposed constraints).

Claims Text - CLTX (6):

(e) the client sending to the server play-list edit commands during the transmission of continuous media data from at least one clip in the play-list for editing the play-list including the addition of at least one additional clip to the play-list, and the server receiving the play-list edit commands and in response editing the play-list during the broadcast session without interruption of the transmission of continuous media data from the server to said destination.

Claims Text - CLTX (8):

3. The method as claimed in claim 1, wherein the second command is substantially identical in format to the play-list edit commands.

Claims Text - CLTX (12):

5. The method as claimed in claim 1, wherein the server returns to the client an acknowledgement of each play-list edit command, and the acknowledgement of each play-list edit command includes an edited version of the play-list resulting from the server editing the play-list in accordance with said each play-list edit command.

Claims Text - CLTX (17):

10. The method as claimed in claim 1, wherein the server checks whether or not each of said play-list edit commands specifies a change of continuous media too close to broadcast time to avoid an interruption of transmission of the continuous media data from the server to said destination, and when one of said play-list edit commands is found to specify a change of continuous media too close to broadcast time to avoid an interruption of transmission of the continuous media data from the server to said destination, not editing the play-list in response to said one of said play-list edit commands, and instead returning an error message to the client.

Claims Text - CLTX (20):

(b) the client sending to the server play-list edit commands during the transmission of continuous media data from at least one clip in the play-list for editing the play-list including the addition of at least one additional clip to the play-list, and the server receiving the play-list edit commands and in response editing the play-list during the broadcast session without interruption of the transmission of continuous media data from the server to said destination;

Claims Text - CLTX (21):

wherein the server checks whether or not each of said play-list edit commands specifies a change of continuous media too close to broadcast time to avoid an interruption of transmission of the continuous media data from the server to said destination, and when one of said play-list edit commands is found to specify a change of continuous media too close to broadcast time to avoid an interruption of transmission of the continuous media data from the server to said destination, not editing the play-list in response to said one of said play-list edit commands, and instead returning an error message to the client.

Claims Text - CLTX (22):

12. The method as claimed in claim 11, wherein the server returns to the client an edited version of the play-list when the server edits the play-list in response to each of the play-list edit commands.

Claims Text - CLTX (29):

(c) the client receiving the session handle and in response transmitting to the server at least one play-list edit command including the session handle and specifying at least one continuous media clip to be transmitted from the server to said destination during the broadcast session; and then

Claims Text - CLTX (30):

(d) the server receiving said at least one play-list edit command, producing an edited version of the play-list in accordance with said at least one play-list edit command, and acknowledging the play-list edit command by returning to the client the edited version of the play-list; and then

Claims Text - CLTX (31):

(e) the client receiving the edited version of the play-list and thereafter transmitting a command for beginning the broadcast session; and then

Claims Text - CLTX (33):

(g) the client sending to the server additional play-list edit commands during the transmission of continuous media data from at least one clip in the play-list for editing the play-list including the addition of at least one additional clip to the play-list, and the server receiving the additional play-list edit commands and in response editing the play-list during the broadcast session without interruption of the transmission of continuous media data from the server to said destination.

Claims Text - CLTX (38):

21. The method as claimed in claim 16, wherein the server checks whether or not each of said play-list edit commands specifies a change of continuous media too close to broadcast time to avoid an interruption of transmission of the continuous media data from the server to said destination, and when one of said play-list edit commands is found to specify a change of continuous media too close to broadcast time to avoid an interruption of transmission of the continuous media data from the server to said destination, not editing the play-list in response to said one of said play-list edit commands, and instead returning an error message to the client.

Claims Text - CLTX (45):

(c) receiving from the client play-list edit commands during the transmission of continuous media data from at least one clip in the play-list for editing the play-list including the addition of at least one additional clip to the play-list, and in response editing the play-list during the broadcast session without interruption of the transmission of continuous media data from the server to said destination.

Claims Text - CLTX (47):

24. The continuous media server as claimed in claim 22, wherein the second command is substantially identical in format to the play-list edit commands.

Claims Text - CLTX (49):

26. The continuous media server as claimed in claim 22, wherein the controller is programmed for returning to the client an acknowledgement of each play-list edit command, and the acknowledgement of each play-list edit command includes an edited version of the play-list resulting from the server editing the play-list in accordance with said each play-list edit command.

Claims Text - CLTX (54):

31. The continuous media server as claimed in claim 22, wherein the controller is programmed for checking whether or not each of said play-list edit commands specifies a change of continuous media too close to broadcast time to avoid an interruption of transmission of the continuous media data from the server to said destination, and when one of said play-list edit commands is found to specify a change of continuous media too close to broadcast time to avoid an interruption of transmission of the continuous media data from the server to said destination, for not editing the play-list in response to said one of said play-list edit commands, and instead returning an error message to the client.

Claims Text - CLTX (60):

(b) receiving from the client play-list edit commands during the transmission of continuous media data from at least one clip in the play-list, and in response editing the play-list during the broadcast session, including the addition of at least one additional clip to the play-list, without interruption of the transmission of continuous media data from the server to said destination;

Claims Text - CLTX (61):

wherein the controller is programmed for checking whether or not each of said play-list edit commands specifies a change of continuous media too close to broadcast time to avoid an interruption of transmission of the continuous media data from the server to said destination, and when one of said play-list edit commands is found to specify a change of continuous media too close to broadcast time to avoid an interruption of transmission of the continuous media data from the server to said destination, not editing the play-list in response to said one of said play-list edit commands, and instead returning an error message to the client.

Claims Text - CLTX (62):

33. The continuous media server as claimed in claim 32, wherein the controller is programmed for returning to the client an edited version of the play-list when the controller edits the play-list in response to each of the play-list edit commands.

Claims Text - CLTX (71):

(b) receiving from the client at least one play-list edit command including the session handle and specifying at least one continuous media clip to be transmitted from the server to said destination during the broadcast session, editing a play-list for the session in accordance with the play-list edit command to produce an edited version of the play-list, and returning to the client the edited version of the play-list;

Claims Text - CLTX (73):

(d) receiving from the client additional play-list edit commands during the transmission of continuous media data from at least one clip in the play-list, and in response editing the play-list during the broadcast session without interruption of the transmission of continuous media data from the server to said destination.

Claims Text - CLTX (78):

42. The continuous media server as claimed in claim 37, wherein the controller is programmed for checking whether or not each of said play-list edit commands specifies a change of continuous media too close to broadcast time to avoid an interruption of transmission of the continuous media data from the server to said destination, and

when one of said play-list edit commands is found to specify a change of continuous media too close to broadcast time to avoid an interruption of transmission of the continuous media data from the server to said destination, for not editing the play-list in response to said one of said play-list edit commands, and instead returning an error message to the client.

Claims Text - CLTX (82):

(c) controlling the server for receiving from the client play-list edit commands during the transmission of continuous media data from at least one clip in the play-list, and in response editing the play-list, including the addition of at least one additional clip to the play-list, without interruption of the transmission of continuous media data from the server to said destination during the broadcast session.

Claims Text - CLTX (84):

45. The program storage device as claimed in claim 43, wherein the second command is substantially identical in format to the play-list edit commands.

Claims Text - CLTX (86):

47. The program storage device as claimed in claim 43, wherein the program is executable by the server for returning to the client an acknowledgement of each play-list edit command, and for including, in the acknowledgement of each play-list edit command, an edited version of the play-list resulting from the server editing the play-list in accordance with said each play-list edit command.

Claims Text - CLTX (91):

52. The program storage device as claimed in claim 43, wherein the program is executable by the server for checking whether or not each of said play-list edit commands specifies a change of continuous media too close to broadcast time to avoid an interruption of transmission of the continuous media data from the server to said destination, and when one of said play-list edit commands is found to specify a change of continuous media too close to broadcast time to avoid an interruption of transmission of the continuous media data from the server to said destination, for not editing the play-list in response to said one of said play-list edit commands, and instead returning an error message to the client.

Claims Text - CLTX (94):

(b) controlling the server for receiving from the client play-list edit commands during the transmission of continuous media data from at least one clip in the play-list, and in response editing the play-list during the broadcast session, including the addition of at least one additional clip to the play-list, without interruption of the transmission of continuous media data from the server to said destination;

Claims Text - CLTX (95):

wherein the program is executable by the server for checking whether or not each of said play-list edit commands specifies a change of continuous media too close to broadcast time to avoid an interruption of transmission of the continuous media data from the server to said destination, and when one of said play-list edit commands is found to specify a change of continuous media too close to broadcast time to avoid an interruption of transmission of the continuous media data from the server to said destination, for not editing the play-list in response to said one of said play-list edit commands, and instead returning an error message to the client.

Claims Text - CLTX (96):

54. The program storage device as claimed in claim 53, wherein the program is executable by the server for returning to the client an edited version of the play-list when the controller edits the play-list in response to each of the play-list edit commands.

Claims Text - CLTX (102):

(b) controlling the server for receiving from the client at least one play-list edit command including the session handle and specifying at least one continuous media clip to be transmitted from the server to said destination during the broadcast session, and for editing a play-list for the session in accordance with the play-list edit command to produce an edited version of the play-list, and for returning to the client the edited version of the play-list;

Claims Text - CLTX (104):

(d) controlling the server for receiving from the client additional play-list edit commands during the transmission of continuous media data from at least one clip in the play-list, and in response editing the play-list during the broadcast session without interruption of the transmission of continuous media data from the server to said destination.

Claims Text - CLTX (109):

63. The program storage device as claimed in claim 58, wherein the program is executable by the server for checking whether or not each of said play-list edit commands specifies a change of continuous media too close to broadcast time to avoid an interruption of transmission of the continuous media data from the server to said destination, and when one of said play-list edit commands is found to specify a change of continuous media too close to broadcast time to

avoid an interruption of transmission of the continuous media data from the server to said destination, for not editing the play-list in response to said one of said play-list edit commands, and instead returning an error message to the client.

US-PAT-NO: 5983069

DOCUMENT-IDENTIFIER: US 5983069 A

TITLE: Point of purchase video distribution system

DATE-ISSUED: November 9, 1999

US-CL-CURRENT: 725/67, 725/87

APPL-NO: 08/ 620642

DATE FILED: March 22, 1996

PARENT-CASE:

This is a Continuation of application Ser. No. 08/301,320 filed Sep. 6, 1994, now U.S. Pat. No. 5,566,353.

US Patent No. - PN (1):

5983069

Drawing Description Text - DRTX (8):

FIG. 7 is a process flow chart of the system's Edit Playlist form.

Detailed Description Text - DETX (47):

The Network Management software includes, for example, the following databases: Store Info Database, Playlist Database, Clip Library Database, Template Database, System Database, Equipment Database and Uplink Database. First, the Store Info Database has, for example, (1) a table of store site information (e.g., store ID, store name, street address, city, state, country, country code, area code, exchange, number, manager, modification information, modifying operator, data telephone, etc.), (2) a site disk contents table which lists all the clips located at that particular receiving site, and (3) the number and content of site playlist sockets (playlist sockets are places where a "wheel", as defined above, can be placed).

Detailed Description Text - DETX (48):

The Playlist Database has, for example, (1) a table of playlists names for the receiving sites which includes the playlist IDs along with the template IDs (a template works as the backbone of the playlist and is described in greater detail below), the playlist's create date, the creating operator, the playlist's modification date and the modifying operator, and (2) a contents table for each playlist that gives information on the clips (e.g., clip sequence number) included in each playlist.

Detailed Description Text - DETX (55):

Part of the system's software is divided into forms. These forms perform as subroutines would in a computer program. In the preferred embodiment, a Windows menu is utilized to walk the user through the various options and forms which are available. FIG. 5 is the diagram of a possible Windows menu setup for the Network Management Program. The user will initially see the following five available choices: FILE, EDIT, PLAYLISTS, UPLINK and MODEM. First, FILE allows the user to control the files which are open to that user. Each user has a security level which allows that user access to certain programs. User's with lower security levels will only be allowed to partially configure (i.e., edit rather than create) the playlists which will be played in the stores.

Detailed Description Text - DETX (56):

PLAYLISTS permits the user (depending on that user's security level) to edit a playlist with minor changes, or to create an entire playlist from scratch, or even to create a new template (all playlists are built upon playlist templates to ensure a proper playlist balance). The PLAYLISTS option loads the Socket Management Form 262 or 263 and gives the user access to certain parts of that form depending on the user's security level.

Detailed Description Text - DETX (60):

The user can then select from the available playlists to fill sockets of a particular store site 266. Each store site has its own number of sockets. For example, if a store is displaying 30 minute wheels of playlists for 18 hours (all the store's open hours), the store has 36 sockets which must be filled with wheels of playlists. At this point, the user can also edit a playlist and/or create a new playlist depending on the user's security level. To edit a playlist, the Edit Playlist Form 267 is loaded.

Detailed Description Text - DETX (61):

FIG. 7 is a process flow chart of the system's Edit Playlist Form. When the user selects the Edit Playlist option, the program enters the Edit Playlist Form 267 and displays a list of the contents of the selected playlist, the contents of the template used to construct that playlist, and the contents of the system's clip library (the clips being segregated by clip type) 270. The Template Database, the Playlist Database and the Clip Library Database provide the information needed for this display. The template used to construct the playlist is the backbone of the playlist. The template predefines the order of certain types of selected video clips. For example, a template for a 30 minute wheel playlist may begin with a commercial clip, then a news clip, then a fact clip, then another commercial clip, etc. This ensures the balance of the playlist and avoids any undesired order of certain clip types (e.g., 8 commercials in a row).

Detailed Description Text - DETX (65):

The user can then select the desired playlist template 276. This desired template can be edited or made into a new template. To edit the template, the program loads the Edit Template Form 279 (described below). Templates cannot be edited after they have been finalized (finalizing is described below). To make a new template, the user provides a new template name and the program adds the new template name along with an empty template to the Template Database 277. The user can then add to this empty template through the Edit Template Form 278.

Detailed Description Text - DETX (67):

After the user is done building and modifying the template, the user can choose between saving or finalizing that template. If the user saves the template, the originally selected template is removed from the existing table in the Template Database and replaced with the newly created template 283. If the user finalizes the template, (1) the program informs the user that this is a one-way operation because the template can no longer be changed/edited, and (2) the template becomes usable (a playlist cannot be added to a template until the user has finalized the template) 284. In addition, finalizing removes the selected template from the existing table in the Template Database and replaces the template with the newly finalized information (adding finalized data to the Template Database) 285. Finally, after a user has based a playlist on a template, the user cannot edit the template. If the user decides to change the template after a playlist has been based on it, the user must start with a new template (create a new template).

Detailed Description Text - DETX (68):

As stated above, while in the Socket Management Form, the user can edit a playlist and/or create a new playlist depending on the user's security level. To create a new playlist, the New Playlist Form 268 is loaded. FIG. 10 is a process flow chart of the system's New Playlist Form. After loading the New Playlist Form 268, the program loads the Open Template Form 273 and goes through that form as described above (see FIG. 8). When the user leaves the Open Template Form, the user returns to the New Playlist Form and is asked to input the name of the new playlist along with any comments the user may have 286. When the user is done, the Playlist Database is updated 287 and the Edit Playlist Form is loaded 267 (see FIG. 7). In sum, when creating a new playlist, the user gets an empty template to fill.